ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»  
(РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ  
О ЛАБОРАТОРНОЙ РАБОТЕ №3.3

По дисциплине «Языки программирования»

ВАРИАНТ 14

Выполнил: ст. гр. ТКИ-142

Кузнецов Денис Константинович

Проверил: к.т.н., доц. Васильева М. А.

(Проверил: к.т.н, доц. Балакина Е. П.)

Москва 2024

1. Формулировка задания

Протабулировать заданную функцию и сумму функционального ряда разложения этой функции на интервале [*a,b*] и с шагом *h* (шаг и интервал задается в константах). Функциональнй ряд вычисляется по соответствующей рекуррентной формуле с заданной точностью *ɛ*. В результате показать три столбца: значение аргумента, значение функции в данной точке и значение суммы ряда, вычисленное с заданной точностью в данной точке. Два последних столбца должны иметь близкие результаты.

Таблица 1 – Формулировка задания 3.3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вар | Функция y | Сумма | Интервал | ɛ |
| 14 |  |  |  | 20-4 |

1. Блок-схема алгоритма

Блок-схема основного алгоритма представлена ниже (Рисунок 1). Блок-схемы функций расчета значений функции, последовательности и ввода представлены ниже (Рисунок 2, 3, 4, 5, 6, 7).



Рисунок 1 ­ Блок-схема основного алгоритма



Рисунок 2 – Блок-схема функции input()



Рисунок 3 – Блок-схема функции intervalCheck(iStart, iEnd)



­Рисунок 4 – Блок-схема функции checkStep(step)



Рисунок 5 – Блок-схема функции getFunc(a)



Рисунок 6 – Блок-схема функции nextElement(a, n)



Рисунок 7 – Блок-схема функции getSum(x, eps)

1. Текст программы на языке C

#include <stdio.h>

#include <stdlib.h>

#include <errno.h>

#include <math.h>

#include <float.h>

/\*\*

\* @brief проверка введенного значения

\* @return возвращает значение

\*/

double input(void);

/\*\*

\* @brief проверка значений интервала

\* @param iStart начало интервала

\* @param iEnd конец интервала

\* @return ошибка, если интервал некорректный

\*/

void intervalCheck(const double iStart, const double iEnd);

/\*\*

\* @brief проверяет значение шага

\* @param step значение шага

\* @return ошибка, если шаг некорректный

\*/

void stepCheck(const double step);

/\*\*

\* @brief Вызвает функцию y = (e^a - e^(-a)) / 2

\* @param a значение переменной a

\* @return Возвращает значение функции

\*/

double getFunc(const double a);

/\*\*

\* @brief Рассчитывает значение следующего элемента ряда

\* @param a значение переменной a

\* @param n индекс элемента последовательности

\* @return Возвращает значение следующего элемента ряда

\*/

double nextElement(const double a, const int n);

/\*\*

\* @brief Рассчитывает значение суммы ряда с точностью доepsilon

\* @param a значение переменной a

\* @param eps значение величины точности

\* @return Возвращает значение суммы ряда

\*/

double getSum(const double a, const double eps);

/\*\*

\* @brief Точка входа в программу

\* @return Возвращает 0 в случае успеха

\*/

int main(void)

{

const double eps = pow(20, -4);

printf("Введите начало интервала: ");

const double iStart = input();

printf("Введите конец интервала: ");

const double iEnd = input();

intervalCheck(iStart, iEnd);

printf("Введите значение шага: ");

const double step = input();

stepCheck(step);

printf("%-10s %-15s %-15s\n", "a", "f(a)", "S(a)");

for (double a = iStart; a <= iEnd + DBL\_EPSILON; a += step) {

printf("%-6.2f %-10.6f %-10.6f\n", a, getFunc(a), getSum(a, eps));

}

return 0;

}

double input(void) {

double value = 0.0;

int result = scanf("%lf", &value);

if (result != 1) {

errno = EIO;

perror("Ошибка ввода!");

exit(EXIT\_FAILURE);

}

return value;

}

void intervalCheck(const double iStart, const double iEnd) {

if (iEnd <= iStart) {

errno = EINVAL;

perror("Неверно введён интервал");

exit(EXIT\_FAILURE);

}

}

void stepCheck(const double step) {

if (step <= DBL\_EPSILON) {

errno = EINVAL;

perror("Неверно введен шаг или его (шага) значение слишком мало");

exit(EXIT\_FAILURE);

}

}

double getFunc(const double a) {

return (exp(a) - exp(-a)) / 2.0;

}

double nextElement(const double a, const int n) {

return (a \* a) / ((2 \* n) \* (2 \* n + 1));

}

double getSum(const double a, const double eps) {

double element = getFunc(a);

double sum = 0.0;

int n = 0;

while (fabs(element) > eps) {

sum += element;

element \*= nextElement(a, ++n);

}

return sum;

}

1. Результаты выполнения программы

Результаты выполнения программы представлены ниже (Рисунок 8, 9, 10, 11, 12, 13).

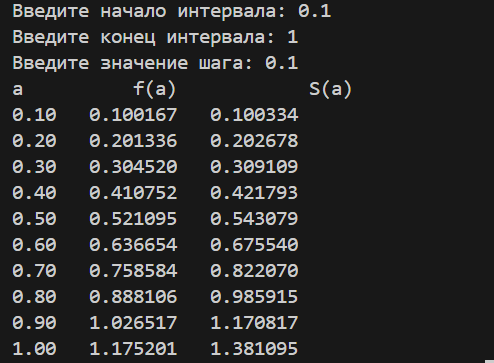


Рисунок 8 – Результаты выполнения программы



Рисунок 9 – Вывод, когда начало интервала является буквой

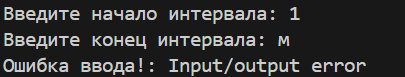


Рисунок 10 – Вывод, когда конец интервала является буквой

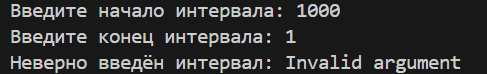


Рисунок 11 – Вывод, когда конец интервала меньше начала

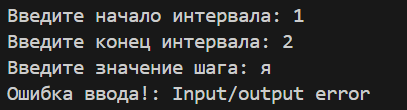


Рисунок 12 – Вывод, когда введенный шаг - буква

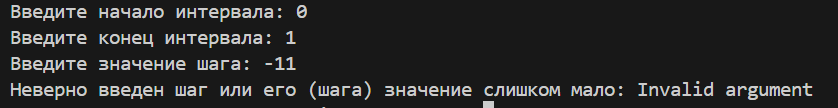


Рисунок 13 – Вывод, когда шаг является отрицательным

1. Выполнение тестовых примеров

На языке Python выполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 14).

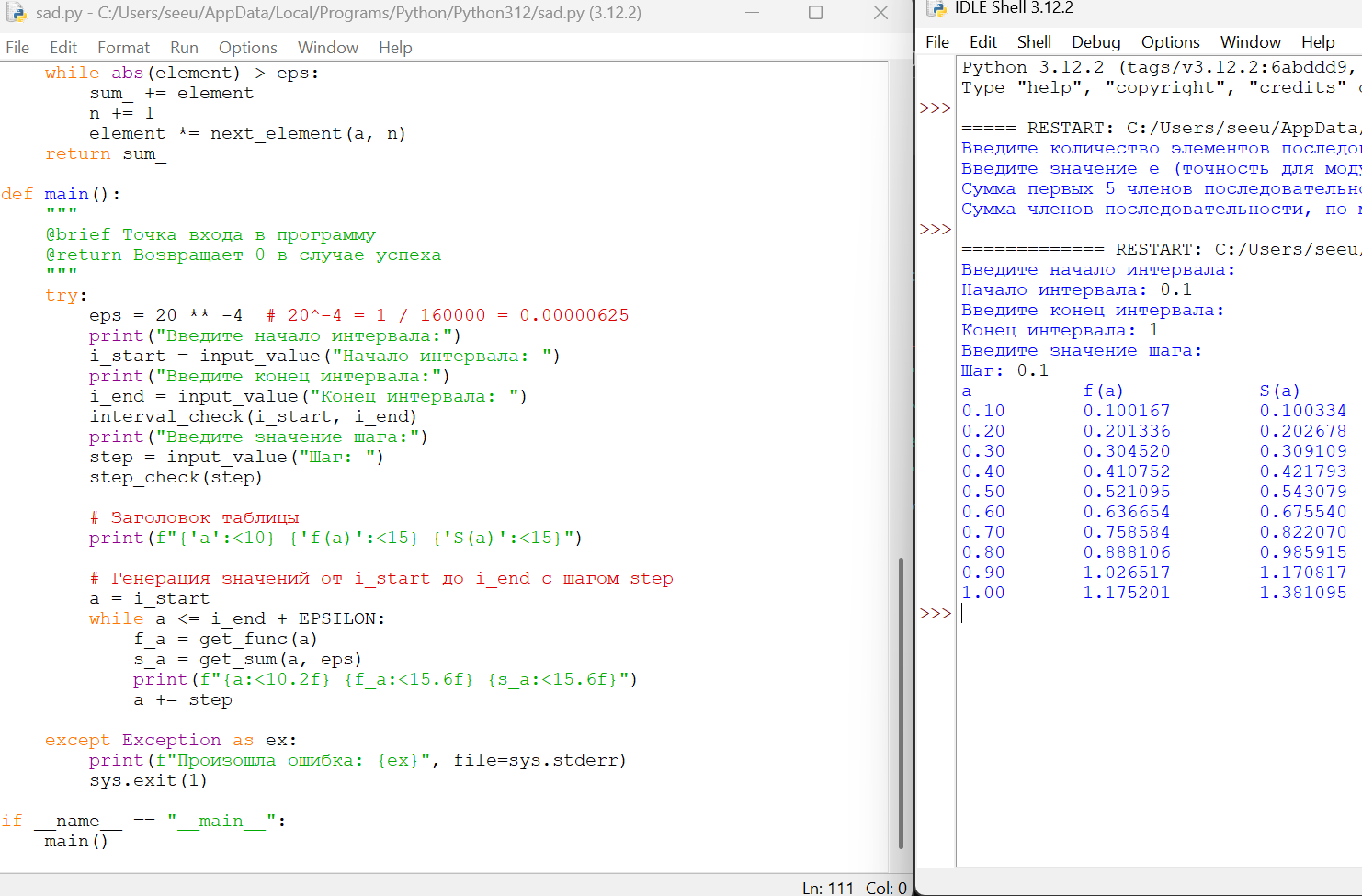


Рисунок 14 – Выполнение программы на языке Python

1. Отметка о выполнении задания в веб-хостинге системы контроля версий

